

07-03-00

A

UTILITY PATENT APPLICATION TRANSMITTAL

(Large Entity)

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Docket No.
13768.143

Total Pages in this Submission
3

TO THE ASSISTANT COMMISSIONER FOR PATENTS

Box Patent Application
Washington, D.C. 20231

Transmitted herewith for filing under 35 U.S.C. 111(a) and 37 C.F.R. 1.53(b) is a new utility patent application for an invention entitled:

METHODS AND SYSTEMS FOR PREVENTING SOCKET FLOODING DURING DENIAL OF SERVICE ATTACKS

and invented by:

Bilal Alam and Michael Courage

If a **CONTINUATION APPLICATION**, check appropriate box and supply the requisite information:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No.: _____

Which is a:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No.: _____

Which is a:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No.: _____

X Application claims priority to US provisional application serial number 60/189,096 filed 14 March 2000

Enclosed are:

Application Elements

1. ☒ Filing fee as calculated and transmitted as described below
2. ☒ Specification having 28 pages and including the following:
 - a. ☒ Descriptive Title of the Invention
 - b. ☒ Cross References to Related Applications (if applicable)
 - c. ☐ Statement Regarding Federally-sponsored Research/Development (if applicable)
 - d. ☐ Reference to Microfiche Appendix (if applicable)
 - e. ☒ Background of the Invention
 - f. ☒ Brief Summary of the Invention
 - g. ☒ Brief Description of the Drawings (if drawings filed)
 - h. ☒ Detailed Description
 - i. ☒ Claim(s) as Classified Below
 - j. ☒ Abstract of the Disclosure

UTILITY PATENT APPLICATION TRANSMITTAL
(Large Entity)

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Docket No.
13768.143

Total Pages in this Submission
3

Application Elements (Continued)

3. ☒ Drawing(s) *(when necessary as prescribed by 35 USC 113)*
- a. ☒ Formal Number of Sheets 5
- b. ☐ Informal Number of Sheets _____
4. ☐ Oath or Declaration
- a. ☐ Newly executed *(original or copy)* ☐ Unexecuted
- b. ☐ Copy from a prior application (37 CFR 1.63(d)) *(for continuation/divisional application only)*
- c. ☐ With Power of Attorney ☐ Without Power of Attorney
- d. ☐ DELETION OF INVENTOR(S)
Signed statement attached deleting inventor(s) named in the prior application,
see 37 C.F.R. 1.63(d)(2) and 1.33(b).
5. ☐ Incorporation By Reference *(usable if Box 4b is checked)*
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under
Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby
incorporated by reference therein.
6. ☐ Computer Program in Microfiche *(Appendix)*
7. ☐ Nucleotide and/or Amino Acid Sequence Submission *(if applicable, all must be included)*
- a. ☐ Paper Copy
- b. ☐ Computer Readable Copy *(identical to computer copy)*
- c. ☐ Statement Verifying Identical Paper and Computer Readable Copy

Accompanying Application Parts

8. ☐ Assignment Papers *(cover sheet & document(s))*
9. ☐ 37 CFR 3.73(B) Statement *(when there is an assignee)*
10. ☐ English Translation Document *(if applicable)*
11. ☐ Information Disclosure Statement/PTO-1449 ☐ Copies of IDS Citations
12. ☐ Preliminary Amendment
13. ☒ Acknowledgment postcard
14. ☒ Certificate of Mailing
- ☐ First Class ☒ Express Mail *(Specify Label No.):* EL624147293US

UTILITY PATENT APPLICATION TRANSMITTAL
(Large Entity)

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Docket No.
13768.143

Total Pages in this Submission
3

Accompanying Application Parts (Continued)

15. ☐ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. ☐ Additional Enclosures (please identify below):

Fee Calculation and Transmittal

CLAIMS AS FILED

For	#Filed	#Allowed	#Extra	Rate	Fee
Total Claims	25	- 20 =	5	x \$18.00	\$90.00
Indep. Claims	3	- 3 =	0	x \$78.00	\$0.00
Multiple Dependent Claims (check if applicable) <input type="checkbox"/>					\$0.00
BASIC FEE					\$690.00
OTHER FEE (specify purpose)					\$0.00
TOTAL FILING FEE					\$780.00

- ☒ A check in the amount of **\$780.00** to cover the filing fee is enclosed.
- ☒ The Commissioner is hereby authorized to charge and credit Deposit Account No. **23-3178** as described below. A duplicate copy of this sheet is enclosed.
- ☐ Charge the amount of _____ as filing fee.
- ☒ Credit any overpayment.
- ☒ Charge any additional filing fees required under 37 C.F.R. 1.16 and 1.17.
- ☐ Charge the issue fee set in 37 C.F.R. 1.18 at the mailing of the Notice of Allowance, pursuant to 37 C.F.R. 1.311(b).



Signature

Dated: **June 30, 2000**

Adrian J. Lee
Attorney for Applicant
Registration No. 42,785
WORKMAN, NYDEGGER & SEELEY
1000 Eagle Gate Tower
60 East South Temple
Salt Lake City, Utah 84111

CC:

CERTIFICATE OF MAILING BY "EXPRESS MAIL" (37 CFR 1.10)Applicant(s): **Bilal Alam and Michael Courage**

Docket No.

13768.143

Serial No.

Filing Date

Examiner

Group Art Unit

Invention:

METHODS AND SYSTEMS FOR PREVENTING SOCKET FLOODING DURING DENIAL OF SERVICE ATTACKS

I hereby certify that this Transmittal letter (in duplicate) (*and other documents)
(Identify type of correspondence)



is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under
37 CFR 1.10 in an envelope addressed to: The Assistant Commissioner for Patents, Washington, D.C. 20231 on
June 30, 2000
(Date)

Mandy Jensen

(Typed or Printed Name of Person Mailing Correspondence)

(Signature of Person Mailing Correspondence)EL624147293US

("Express Mail" Mailing Label Number)

Note: Each paper must have its own certificate of mailing.

- * Patent Application (28pgs)
- 5 Sheets of Formal Drawings
- Check No. 116393 in the amount of \$780.00
- Postcard

METHODS AND SYSTEMS FOR PREVENTING SOCKET FLOODING DURING DENIAL OF SERVICE ATTACKS

Variable	Mean	SD	Min	Max
Age	31.2	5.1	18	45
Gender	1.2	0.4	1	2
Marital status	1.1	0.3	1	2
Education	12.5	1.2	10	16
Income	1.5	0.8	1	3
Occupation	1.2	0.4	1	2
Religion	1.1	0.3	1	2
Health status	1.2	0.4	1	2
Stress level	1.5	0.6	1	3
Life satisfaction	1.8	0.5	1	3
Self-esteem	1.6	0.4	1	3
Resilience	1.4	0.5	1	3
Optimism	1.7	0.4	1	3
Gratitude	1.9	0.3	1	3
Forgiveness	1.6	0.4	1	3
Empathy	1.8	0.3	1	3
Compassion	1.7	0.4	1	3
Kindness	1.9	0.3	1	3
Generosity	1.6	0.4	1	3
Patience	1.8	0.3	1	3
Humility	1.7	0.4	1	3
Modesty	1.6	0.4	1	3
Shyness	1.5	0.5	1	3
Introversion	1.4	0.6	1	3
Extroversion	1.6	0.4	1	3
Sociability	1.7	0.3	1	3
Friendliness	1.8	0.3	1	3
Warmth	1.9	0.3	1	3
Openness	1.7	0.4	1	3
Conscientiousness	1.6	0.4	1	3
Neuroticism	1.5	0.5	1	3
Agreeableness	1.8	0.3	1	3
Emotional stability	1.7	0.4	1	3
Psychological well-being	1.9	0.3	1	3
Life purpose	1.8	0.4	1	3
Meaning in life	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	3
Humanistic well-being	1.7	0.4	1	3
Existential well-being	1.6	0.4	1	3
Transcendental well-being	1.5	0.5	1	

WORKMAN, NYDEGGER & SEELEY
A PROFESSIONAL CORPORATION
ATTORNEYS AT LAW
1000 EAGLE GATE TOWER
60 EAST SOUTH TEMPLE
SALT LAKE CITY, UTAH 84111

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24

2

3
4
5

7

8
9
10

12

3
4
5
6
7
8

9
20
21
22
23

Therefore, what are desired are methods and systems for reducing the incidence of service denials due to an attack in which requests are repeatedly made to the server without transmitting any request data.

should in itself be relatively rare. Secondly, even though the backlog queue is full, the period of time between the time a connection is made and the time the data is received is relatively brief for a legitimate connection request. Thus, the chance that the legitimate connection request would be executing in that brief period is also relatively small.

Notwithstanding this small risk, the method may be further optimized to reduce the chances for denying legitimate connection requests even further by allowing the systems administrator to specifying a grace period between the time the backlog queue is determined to be used and the time the identified connection sockets are disconnected. If, during this grace period, the server is able to handle the connection requests in the backlog queue, no connection sockets will be disconnected.

Additional features and advantages of the invention will be set forth in the description which follows, and in part will be obvious from the description, or may be learned by the practice of the invention. The features and advantages of the invention may be realized and obtained by means of the instruments and combinations particularly pointed out in the appended claims. These and other features of the present invention will become more fully apparent from the following description and appended claims, or may be learned by the practice of the invention as set forth hereinafter.

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24

Figure 1 illustrates an exemplary system that provides a suitable operating environment for the present invention;

Figure 3 illustrates a server-implemented process for responding to requests;

Figure 5 illustrates a server-implemented method of protecting against or at least reducing the impact of denial of service attacks.

1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8
9
0
1
2
3
4

2
3
4
5

6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

23
24

implemented. Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules, being executed by computers in network environments. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of the program code means for executing steps of the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represent examples of corresponding acts for implementing the functions described in such steps.

Those skilled in the art will appreciate that the invention may be practiced in network computing environments with many types of computer system configurations, including personal computers, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by local and remote processing devices that are linked (either by hardwired links, wireless links, or by a combination of hardwired or wireless links) through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

With reference to Figure 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a conventional computer 120, including a processing unit 121, a system memory 122, and a system bus 123 that couples various system components including the system memory 122 to the processing unit 121. The system bus 123 may be any of several types of bus structures including a memory bus

or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 124 and random access memory (RAM) 125. A basic input/output system (BIOS) 126, containing the basic routines that help transfer information between elements within the computer 120, such as during start-up, may be stored in ROM 124.

The computer 120 may also include a magnetic hard disk drive 127 for reading from and writing to a magnetic hard disk 139, a magnetic disk drive 128 for reading from or writing to a removable magnetic disk 129, and an optical disk drive 130 for reading from or writing to removable optical disk 131 such as a CD-ROM or other optical media. The magnetic hard disk drive 127, magnetic disk drive 128, and optical disk drive 130 are connected to the system bus 123 by a hard disk drive interface 132, a magnetic disk drive-interface 133, and an optical drive interface 134, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer-executable instructions, data structures, program modules and other data for the computer 120. Although the exemplary environment described herein employs a magnetic hard disk 139, a removable magnetic disk 129 and a removable optical disk 131, other types of computer readable media for storing data can be used, including magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, RAMs, ROMs, and the like.

Program code means comprising one or more program modules may be stored on the hard disk 139, magnetic disk 129, optical disk 131, ROM 124 or RAM 125, including an operating system 135, one or more application programs 136, other program modules 137, and program data 138. A user may enter commands and information into the computer 120 through keyboard 140, pointing device 142, or other input devices (not shown), such as a microphone, joy stick, game pad, satellite dish, scanner, or the like.

When used in a LAN networking environment, the computer 120 is connected to the local network 151 through a network interface or adapter 153. When used in a WAN networking environment, the computer 120 may include a modem 154, a wireless link, or other means for establishing communications over the wide area network 152, such as the Internet. The modem 154, which may be internal or external, is connected to the system bus 123 via the serial port interface 146. In a networked environment, program modules

Figure 2 illustrates a requesting client computer system 210 (hereinafter, “a client”) and a responding server computer system 220 (hereinafter, “a server”) which communicate over a network 230. In a typical request/response communication protocol such as HyperText Transport Protocol (“HTTP”), the client 210 transmits a connection request 240 to the server 220 over the network 230. The server 220 then provides a connection in response to the connection request and transmits a connection confirmation message 250 back to the client 210. The client 210 then transmits request data 260 to the server 220. The request data 260 includes information helpful in identifying what the request is as well as information helpful in fulfilling the request. If appropriate for the request, the server 220 then transmits a response 270 back to the client 210 over the network 230.

The server computer system 220 is a “server” computer system in that it provides a service in the form of a connection and a response to the client computer system 210. The server may also obtain the services of other computer systems over the network. In this context, the server 220 may also be a client computer system. The client computer system 210 is a “client” computer system in that it is served by the server providing the connection and generating the response. The client computer system 210 may provide services to yet other computer systems. In this context, the client computer system may also be a server computer system. The client 210 and the server 220 may each be structure similar to the computer 120 or may contain a subset or superset of the elements described above for the computer 120.

On the other hand, if the server 220 is unable to handle the connection request (“No” in decision block 340), then the connection request is placed in a backlog queue for future handling (step 350). As shown in Figure 4, each listen socket 420A through 420H

has a corresponding backlog queue 430A through 430H. If the server cannot handle the connection request, the connection request is passed into the queue corresponding to the listen socket that the connection request mapped to in step 330. Although each listen socket has a request queue in Figure 4, in an alternative embodiment, a more general backlog queue may be shared between one or more or all of the listen sockets. In this alternative, the server computer system may map the request to the listen socket after the connection request is drawn from the backlog queue during future processing.

The method 300 will now be explained in the context of a WINDOWS® operating system using a Winsock module to establish connections. For each detected connection request, the Winsock module maps the connection request to a listen socket (step 330). To establish a connection to the listen socket, a module may be called that accepts connections and waits for request data before completing. For example, an extension of the Winsock module called Winsock()AcceptEx() is called and the corresponding listen socket is passed in along with the new connection socket that represents the connection to the listen socket. The Winsock()AcceptEx() is completed when request data is beginning to be received from the network in step 370.

Winsock may allocate a pool having a fixed number of Winsock()Accept() calls available for creating new connections. If the entire pool of Winsock()Accept() calls are already processing new connections, then the server is not currently able to satisfy subsequent connection requests (“No” in decision block 340). In this case, the connection request is placed in the backlog queue corresponding to listen socket (step 350).

In normal operation, it should preferably be very rare that the server 220 cannot currently handle a connection request. However, a denial of service attack may often result in the server being unable to currently handle connection requests. In this description and

1 in the claims, a “denial of server attack” is defined as the repetitious transmission of
2 connection requests without a subsequent transmission of request data needed to process
3 the requests. In such a denial of service attack, the method 300 of Figure 3 will proceed
4 through step 360 in which resources are allocated. However, the server does not receive
5 subsequent request data as in step 370. Therefore, the allocated resources are never freed
6 up in step 390. Since connection requests are repeatedly made, the amount of allocated
7 resources rises until the server can no longer allocate resources and thus must deny
8 legitimate requests for service.

9 In the context of the Winsock module, the repeated connection requests will result
10 in repeated calls of the Winsock()AcceptEx() module. However, none of the
11 Winsock()AcceptEx() modules will complete since no request data is sent during a denial
12 of service attack. Thus, the pool of Winsock()AcceptEx() modules will gradually deplete.
13 Eventually, the server 220 will not be able to handle new connection requests, legitimate or
14 not, and the connection requests will be placed in the backlog queue. Eventually, the
15 backlog queue will also be filled up and thus new connection requests will not be saved
16 and thus will never be handled.

17 Figure 5 illustrates a flowchart of a method 500 that prevents or at least reduces the
18 impact of these denial of service attacks. As mentioned above, when the server 220 cannot
19 currently handle a connection request, the connection request is place in a backlog queue.
20 The method 500 monitors this backlog queue (step 510). Accordingly, embodiments
21 within the scope of the present invention include a means and/or step for monitoring the
22 backlog queue. Any method of monitoring the backlog queue will suffice so long as the
23 method is capable of determining whether of not there are entries in the backlog queue. In
24 the example shown in Figure 4, each listen socket has a corresponding backlog queue. The

1 method 500 may monitor these backlog queues by, for example, calling modules that scan
2 the backlog queues to determine usage. On such module is a Winsock extension called
3 Winsock(select). A list of listen sockets is passed into the Winsock(select) function.
4 The Winsock(select) module monitors the backlog queue of each of the listens sockets in
5 the list of listen sockets passed into the Winsock(select) module.

Next, the method 500 determines if the backlog queue is being used (step 520). Any method for determining that the backlog queue is being used will suffice. In the above example where the Winsock(select) extension of Winsock is used to monitor the backlog queue, the determination is made by the very fact that the Winsock(select)extension module returns. The Winsock(select) extension module returns when one or more of the listen sockets have entries in their corresponding backlog queues.

Next, the method 500 resets one or more connection sockets upon notification that the backlog queue is being used (step 530). Accordingly, embodiments within the scope of the present invention include a means and/or step for resetting one or more connection sockets upon notification that the backlog queue is being used.

As part of the step for resetting one or more listen sockets, the method 500 includes a step of determining which connection sockets have established connections, but have not received any data (step 540). In the context of using Winsock, the server computer system 220 enumerates all the connection sockets that have been created using a currently called Winsock()AcceptEx() function. For each of these currently called Winsock()AcceptEx() connection sockets, the extension Winsock()getsockopt() is used to determine whether or not a connection has been established. If a connection has been established, then the connection socket is suspected of being caused by a malicious connection request since a

There is some risk associated with closing a connection socket simply because it has a connection but no received request data. For example, the connection socket may not have been created as a result of a malicious connection request. Instead, it may be the connection request was legitimate in that the associated connection socket just happened to be in a stage where the connection was just made but the soon to arrive request data simply has not arrived yet. In this case, a legitimate connection request would be denied.

Notwithstanding this small risk, the method may be further optimized to reduce the chances for denying legitimate connection requests even further. For example, the server computer system 220 may be configured to allow for a specified grace period after entries

The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24

9. The method in accordance with Claim 8, wherein determining that the backlog queue is being used comprises detecting that the module that scans at least the backlog queue has returned.

1 10. The method in accordance with Claim 8, wherein the module that scans at
2 least the backlog queue for activity comprises a Winsock(select()) module.

3
4 11. The method in accordance with Claim 10, wherein determining that the
5 backlog queue is being used comprises detecting that the Winsock(select()) module has
6 returned.

7
8 12. The method in accordance with Claim 1, wherein resetting one or more
9 connection sockets upon notification that the backlog queue is being used comprises the
10 following:

11 identifying any connection sockets that have connections but no received
12 request data; and

13 disconnecting the identified connection sockets.

14
15 13. The method in accordance with Claim 12, wherein identifying any
16 connection sockets that have connections but no received request data comprises the
17 following:

18 calling a module that identifies the state of the connection socket.

19
20
21 14. The method in accordance with Claim 13, wherein the module that
22 identifies the state of the connection socket comprises a Winsock(getsockopt()) module.

23
24 15. The method in accordance with Claim 1, further comprising:

16. The method in accordance with Claim 1, wherein attempting a connection for each connection request received by the server computer system from said one or more client computer systems comprises establishing a connection.

17. A computer program product for use in a networked environment, wherein one or more client computer systems make requests for information from a server computer system, the server computer system providing information in response to the requests from the one or more client computer systems, the server computer system having one or more listen sockets and having a backlog queue for queuing connection requests that the server computer system cannot currently handle, a computer program product for implementing a method of reducing denials of service even though the server computer system is experiencing a denial of service attack, wherein the computer program product comprises computer-executable instructions which, when executed by a processor, implements the following:

attempting a connection for each connection request received by the server computer system from said one or more client computer systems;

for each connection request that the server computer system cannot currently handle, placing the connection request in a backlog queue;

monitoring the backlog queue;

determining that the backlog queue is being used;

resetting one or more connection sockets upon notification that the backlog queue is being used.

18. The computer program product in accordance with Claim 17, further comprising computer-executable instructions for mapping each connection request to a corresponding listen socket, wherein each listen socket has a corresponding backlog queue.

specifying a grace period between the time the backlog queue is determined to be used and the time one or more connection sockets are reset to allow the server computer system to empty the backlog queue, wherein the resetting of the one or

000530" 005/0560

1 more connection sockets is performed only if the backlog queue still has entries
2 after the grace period.

3
4 23. The computer program product in accordance with Claim 17, wherein the
5 computer-executable instructions for attempting a connection for each connection request
6 received by the server from said one or more clients comprise computer-executable
7 instructions for establishing a connection.

25. The method in accordance with Claim 24, further comprising:
specifying a grace period between the time the backlog queue is determined
to be used and the time the identified connection sockets are disconnected, wherein

the disconnection is performed only if the backlog queue still has entries after the grace period.

WORKMAN, NYDEGGER & SEELEY

Table 1. Demographic characteristics of the study population	
Age (years)	50.0 ± 10.0
Gender	
Male	50.0%
Female	50.0%
Education level	
High school	50.0%
University	50.0%
Occupation	
White collar	50.0%
Blue collar	50.0%
Unemployed	50.0%
Marital status	
Married	50.0%
Single	50.0%
Divorced	50.0%
Widowed	50.0%
Health status	
Good	50.0%
Fair	50.0%
Poor	50.0%
Smoking status	
Smoker	50.0%
Non-smoker	50.0%
Alcohol consumption	
Drinker	50.0%
Non-drinker	50.0%
Family size	
1-2	50.0%
3-4	50.0%
5-6	50.0%
7-8	50.0%
9-10	50.0%
11-12	50.0%
13-14	50.0%
15-16	50.0%
17-18	50.0%
19-20	50.0%
21-22	50.0%
23-24	50.0%
25-26	50.0%
27-28	50.0%
29-30	50.0%
31-32	50.0%
33-34	50.0%
35-36	50.0%
37-38	50.0%
39-40	50.0%
41-42	50.0%
43-44	50.0%
45-46	50.0%
47-48	50.0%
49-50	50.0%
51-52	50.0%
53-54	50.0%
55-56	50.0%
57-58	50.0%
59-60	50.0%
61-62	50.0%
63-64	50.0%
65-66	50.0%
67-68	50.0%
69-70	50.0%
71-72	50.0%
73-74	50.0%
75-76	50.0%
77-78	50.0%
79-80	50.0%
81-82	50.0%
83-84	50.0%
85-86	50.0%
87-88	50.0%
89-90	50.0%
91-92	50.0%
93-94	50.0%
95-96	50.0%
97-98	50.0%
99-100	50.0%
101-102	50.0%
103-104	50.0%
105-106	50.0%
107-108	50.0%
109-110	50.0%
111-112	50.0%
113-114	50.0%
115-116	50.0%
117-118	50.0%
119-120	50.0%
121-122	50.0%
123-124	50.0%
125-126	50.0%
127-128	50.0%
129-130	50.0%
131-132	50.0%
133-134	50.0%
135-136	50.0%
137-138	50.0%
139-140	50.0%
141-142	50.0%
143-144	50.0%
145-146	50.0%
147-148	50.0%
149-150	50.0%
151-152	50.0%
153-154	50.0%
155-156	50.0%
157-158	50.0%
159-160	50.0%
161-162	50.0%
163-164	50.0%
165-166	50.0%
167-168	50.0%
169-170	50.0%
171-172	50.0%
173-174	50.0%
175-176	50.0%
177-178	50.0%
179-180	50.0%
181-182	50.0%
183-184	50.0%
185-186	50.0%
187-188	50.0%
189-190	50.0%
191-192	50.0%
193-194	50.0%
195-196	50.0%
197-198	50.0%
199-200	50.0%
201-202	50.0%
203-204	50.0%
205-206	50.0%
207-208	50.0%
209-210	50.0%
211-212	50.0%
213-214	50.0%
215-216	50.0%
217-218	50.0%
219-220	50.0%
221-222	50.0%
223-224	50.0%
225-226	50.0%
227-228	50.0%
229-230	50.0%
231-232	50.0%
233-234	50.0%
235-2	

Docket No. 13768.143

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24

G:\DATA\PAT\WORDPAT\13768143.doc

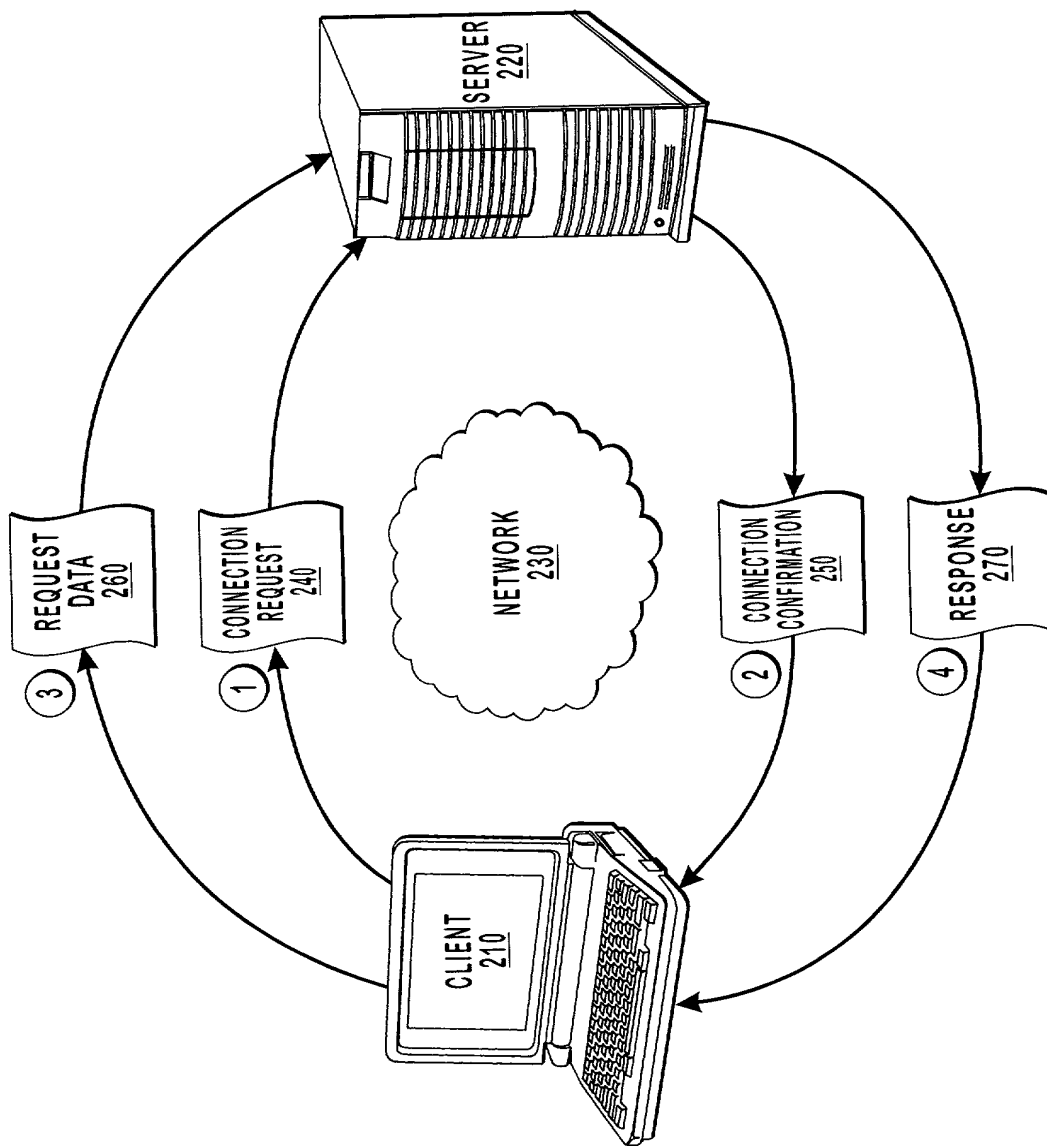


FIG. 2

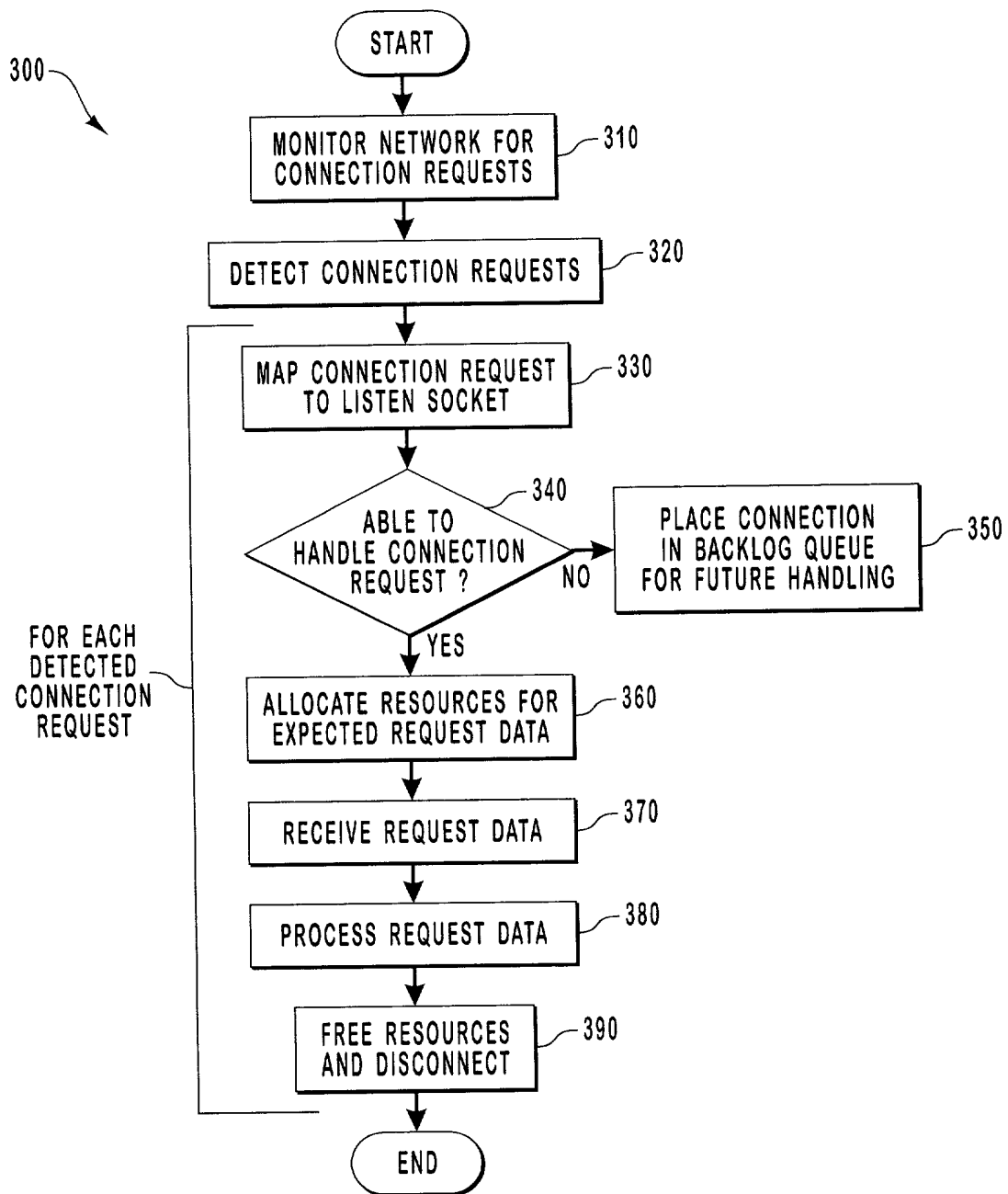


FIG. 3

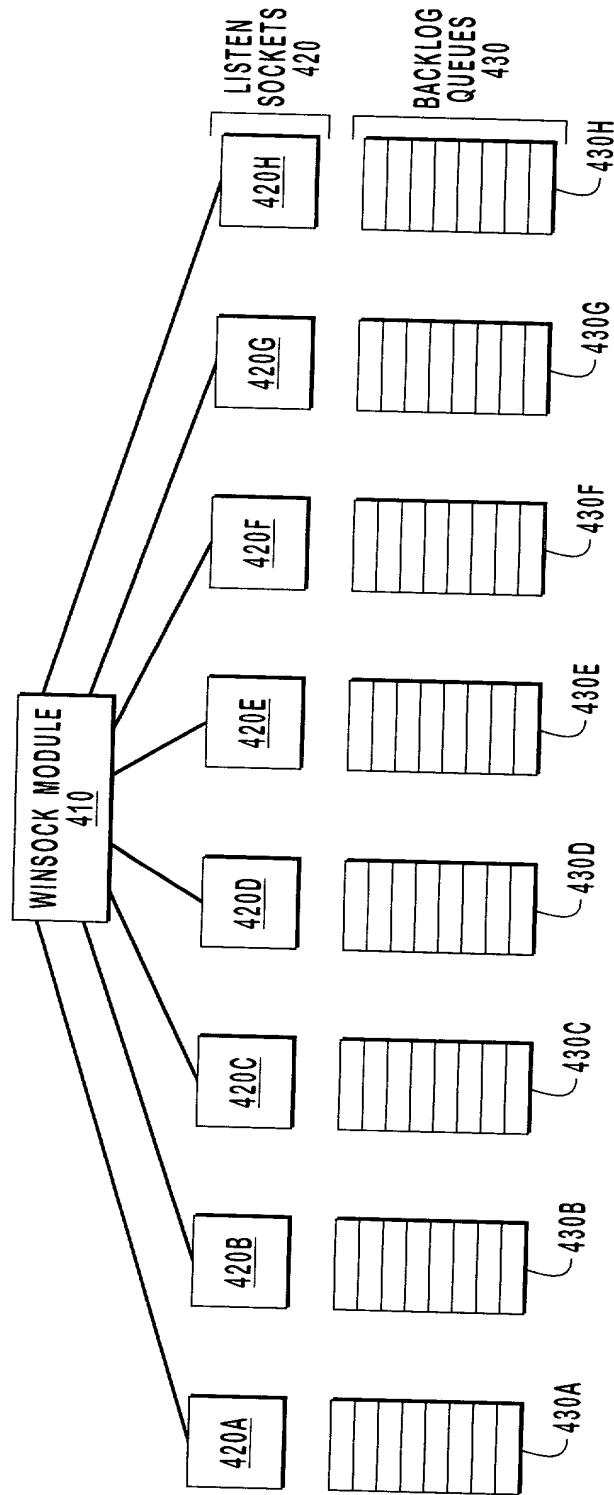


FIG. 4

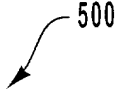


FIG. 5